A MULTI-REWARD STOCHASTIC MODEL FOR THE COMPLETION
TIME OF PARALLEL TASKS *

Andrea BOBBIO

Istituto Elettrotecnico Nazionale Galileo Ferraris
Strada delle Cacce 91, 10135 Torino, Italy

This paper deals with the analysis of the completion time of parallel tasks executed on a system
experiencing random variation of its performance capacity versus time. The time behavior of
the system configuration is modelled by a homogeneous Markov chain, and the effective rate of
service of each task is represented by a vector of reward rates superimposed to the structure-
state process. This unified framework forms a Multi-Reward Markov Model MRMM. The
paper discusses the modelling capabilities of MRMM with reference to a simple example.

1 Introduction

This paper considers the problem of computing the
distribution of the completion time of tasks which are
simultaneously running on a distributed system that
is subjected to random variations in its computation
capacity due to failures and repairs. The system be-


behavior in time is described by a stochastic process re-
ferred to as the structure state process. Even if more
complex formulations are possible [1], the structure
state process is restricted to be a Markov chain.

The system executes parallel tasks, and each task
is characterized by its work requirement. In gen-


eral, the work requirement is a random variable with


known distribution. In each configurational state a


vector of reward rates is defined, whose entries repre-


sent the rate of service of each task. This comprehe-


nsive framework forms a Multi-Reward Markov Model


(MRMM).

A task is completed when the computational time
accumulated by the system in all the visited structure-


states equals the work requirement. For each task, the


reward can be accumulated according to two possible


policies called preemptive resume (prs) and preemptive


repeat (prd), respectively [2]. Mixed policies are also


possible: a single structure-state can behave as prs


with respect to a partition of the tasks and prd with


respect to the complementary partition of the tasks.

The analysis of a pure accumulation (prs) process


with a single reward rate has been previously con-


sidered in [3,4,5,2,6]. A prd accumulation mechanism


originates from studies of queueing systems with a fail-


ure prone server [7], and has been discussed in [2] when


the underlying process is a Markov chain, and in [6]


for a semi-Markov process.

We generalize previous work in this area by consid-


ering parallel tasks with different work requirements
to be executed simultaneously on the system with ser-


vice rates which are state dependent and task depend-


ent. This unified dependability model can accommodate
different disciplines of load sharing, different pre-


emption and load redistribution policies, by properly


modifying the reward rates associated to the states of


the structure state process.

In Section 2, the completion time problem is for-


mulated, while in Section 3 a physical interpretation


in the area of the performability analysis of degrad-


able systems is given. A numerical example concludes


the paper in Section 6.

2 The completion time

Let the structure-state process Z(t) be a Markov chain
with cardinality n, infinitesimal generator Q = [qij]
and initial probability vector $P^0$.

The system executes $\nu$ tasks in parallel. To avoid


confusion, we will use latin letters ($i,j, \ldots$) for pedices


indexing states of $Z(t)$, and greek letters ($\alpha,\beta,\gamma,\ldots$)


for pedices indexing tasks. Each task $\alpha$ ($\alpha = 1, \ldots, \nu$)
is served in each state $i$ ($i = 1, \ldots, n$) at a possibly
different service rate defined by a non-negative reward


rate $r_{\alpha i}$. By this definition, $r_{\alpha i} dt$ is the amount of work


performed on task $\alpha$ given that the system is in state


$i$ during $t - t + dt$. We group the reward rates into a


$n \times \nu$ matrix $R = [r_{\alpha i}]$ ($r_{\alpha i} \geq 0$).

The vector $\mathbf{r} = [r_{1\alpha}, r_{2\alpha}, \ldots, r_{n\alpha}]^T$ is the $ \alpha - \text{th}$


column of $R$; its entries represent the service rate of


*This work was partially supported by the Italian National
Research Council (CNR) under the project "Materials and De-


vices for Solid State Electronics", Grant 89.01857.81.
task \( \alpha \) in each structure-state. Given that the total system capacity available in state \( i \) is:

\[
r_i = \sum_{\alpha=1}^{\nu} r_{i\alpha},
\]

the \( i \)-th row of \( R \) indicates how the system processing capacity \( r_i \) is distributed among the tasks.

The structure state process \( Z(t) \), together with the reward rate matrix \( R \), identifies the Multi-Reward Markov Model (MRMM).

We now define a \( \nu \)-dimensional vector \( B(t) \). Each entry \( B_\alpha(t) \), \( (\alpha = 1, \ldots, \nu) \), represents the accumulation of the \( \alpha \)-th reward rate during the interval \( (0, t) \). \( B(t) \) is a multidimensional stochastic process that depends on \( Z(x) \) for \( x \leq t \) [9]. In order to account for different physical mechanisms of accumulation of the reward in real systems, we introduce two types of policies:

**prs policy** - This policy represents a mechanism of pure accumulation: if the generic entry \( B_\alpha(t) \) belongs to this class, is defined as the integral:

\[
B_\alpha(t) = \int_0^t r_{Z(x), \alpha} dx
\]

where \( r_{Z(x), \alpha} = r_{i\alpha} \) when \( Z(x) = i \). When a transition in \( Z(t) \) occurs the of \( B_\alpha(t) \) is resumed in the new state. For this reason, we will refer to this mechanism as *preemptive resume (prs)* accumulation.

**prd policy** - This policy represents the accumulation of the reward during the sojourn of the process in a single state between two subsequent transitions. When a transition occurs in \( Z(t) \), the value of \( B_\alpha(t) \) drops to zero. Thus, at any time \( B_\alpha(t) \) equals the integral of the reward rate from the epoch of the last transition. For this reason we will refer to this mechanism as *preemptive repeat (prd)* accumulation.

Let us now define a \( \nu \)-dimensional vector of work requirements \( X_\alpha \), \( (\alpha = 1, \ldots, \nu) \). Each entry \( X_\alpha \) is a random variable with distribution \( G_\alpha(x) \). The completion time for task \( \alpha \) is the time at which the accumulated reward \( B_\alpha(t) \) reaches the value \( X_\alpha \) for the first time. Hence, the completion time problem can be modelled as the hitting problem of the multiple stochastic process \( B(t) \) against the vector of absorbing barriers \( X \) [1]. Figure 1 shows a possible realization of a **prs** functional \( B_1(t) \) and of a **prd** functional \( B_2(t) \) defined on the same process \( Z(t) \).

We tag a particular task \( \gamma \) and we introduce the defective random variable \( \gamma T \):

\[
\gamma T = \min \{ t \geq 0 : B_\gamma(t) \geq X_\gamma, \\
B_\alpha(t) < X_\alpha, \ (\alpha = 1, \ldots, \nu; \ \alpha \neq \gamma) \}.
\]

\( \gamma T \) is the minimal completion time when the functional component \( B_\gamma(t) \) is the first one to reach the value \( X_\gamma \). The defective distribution function of \( \gamma T \) is denoted by:

\[
\gamma F(t) = \text{Prob} \{ \gamma T \leq t \}.
\]

The multiple Laplace transform of (4) has been derived in closed form in [1]. Moreover, it has been shown in [1] that \( \gamma F(t) \) is a Phase type (PH) distribution [10] when the barrier levels \( X_\alpha \) are also PH. This closure property of PH distributions is an extension of a similar property proved in [11] for a single task, and will be exploited in the numerical computations.

### 3 Model interpretation

A pictorial representation of a **MRMM** is shown in Figure 2. The system is varying in time according to the stochastic process \( Z(t) \), while at time \( t = 0 \) tasks are put in parallel execution with service rates defined in each structural state by the reward rate matrix \( R \). The output measure calculated on the **MRMM** is the distribution of the completion time of a tagged task (Equation 4). The incorporation of a reward model in the structure-state process offers the possibility of considering under a unified framework the degradation of the system and the interaction of the system
degradation with the accomplishment level of each individual task. As usual in performability analysis, the values of the reward rates to be used in the high level representation can be borrowed from more detailed performance studies. Some possible physical interpretations of the reward model are examined.

![Figure 2 - MRMM of a degradable system with parallel tasks](image)

**Load sharing**
The effect of a load sharing discipline [12] can be captured by acting on the values of the execution rates. Assuming a uniform sharing discipline the processing capacity \( r_i \) of the system in state \( i \) (Equation 1) is equally partitioned among the \( \nu \) simultaneously running tasks; thus:

\[
r_{ia} = r_i / \nu \quad (\alpha = 1, 2, \ldots, \nu)
\]

In a load sharing discipline based on a round-robin scheduling, the service rate of a task, in a macroscopic time scale, depends on the length of the work requirement. The effect of this work dependent discipline, can be modelled by assigning to each task a service rate inversely proportional to the expected value of the work requirement. Thus:

\[
r_{ia} = \rho_i / E[X_\alpha] \quad \text{with} \quad \sum_{\alpha=1}^{\nu} (\rho_i / E[X_\alpha]) = r_i
\]

A strictly priority discipline means that in the given structure-state \( i \) only the highest priority task, say task \( \beta \), can be executed. In this case (for \( \alpha = 1, 2, \ldots, \nu ; \alpha \neq \beta \)):

\[
r_{ia} = 0 \quad ; \quad r_i\beta = r_i
\]

The whole system capacity available in state \( i \) is allocated to task \( \beta \).

**Global degradation and redistribution policies**
A state transition in \( Z(t) \) represents a variation in the system configuration. This variation affects the reward model in two ways: as a modification of the total system capacity available to the new state and as a modification of the way in which the total system capacity is allocated to the tasks. Suppose a transition from \( i \) to \( j \) has occurred in \( Z(t) \). If \( \tau_j < \tau_i \) a degradation has occurred (failure), while \( \tau_j > \tau_i \) corresponds to a restoration (repair).

The way in which the load sharing is modified as a consequence of the variation of the total capacity of the available resources is called the redistribution policy. The redistribution policy, at a macroscopic time scale, affects the task response time [13], and is reflected in the MRMM by a proper partition of the reward rates.

**Preemption policy**
The two different accumulation policies considered in Section 2, can be regarded, at the system level, as representing two classical preemption disciplines arising when considering the interaction of the tasks in execution with a change of state in the system [7,2].

- Under a prs accumulation policy, the system keeps memory of the work already done and the tasks in progress are resumed at each transition.
- Under a prd accumulation policy, the tasks in progress are preempted and must be restarted from scratch at each transition. In particular, we assume that the work requirements of the repeated tasks are resampled from the same distribution. This policy will be referred to as preemptive repeat different (prd) [2].

We assume that the preemption policy is a characteristic attached to each task; the set of the tasks can be partitioned into two subsets, one containing the tasks for which the preemption policy is prs, the other one containing the tasks for which the preemption policy is prd. This partitioning models a situation in which the system is fault-tolerant with respect to a class of tasks (e.g. utility tasks) but it is not with respect to other tasks (e.g. user's tasks).

**Steady state versus transient analysis**
The analysis of a MRMM allows us to study how some characteristic measures evolve in time, while the system itself is in equilibrium. In other words, we can observe the system in a temporal window of length \( t \) located at any point in time, and thus evaluate the transient behaviour of some property being \( Z(t) \) in steady state.

Indeed, in the derivation of the completion time distribution, we can assume the steady state probability vector of the structure state process \( Z(t) \) as initial probability vector \( P_0 \).
4 Numerical example

We consider a multiprocessor system composed of \( n \) identical Processing Elements (PE) with failure rate \( \lambda \). Upon occurrence of a failure the system recovers with probability \( c \) (the coverage probability [14,15]) passing into a working state with one fewer operating PE. A recovered failure is repaired with rate \( \mu \) while a unrecovered failure is repaired with rate \( \delta \). The structure state process \( Z(t) \) for such a system is shown in Figure 3 [14]; the label inside each state is the number of working PE's.

\[ \text{Figure 3 - State-transition diagram of a degradable multiprocessor system} \]

We assume that, at an arbitrarily chosen initial time, when \( Z(t) \) has reached its steady state, one task per PE is activated. Thus, the initial probability vector \( P_0 \) to be used in the computation is the solution of the equilibrium equation \( P \cdot Q = 0 \). The computational problem consists in determining the probability that at a generic time \( t \) a given task (say task \( \gamma \)) has completed first. According to the model interpretation of Section 3, in order to completely define the reward matrix the following specifications should be given:

- the total performance degradation model;
- the load sharing policy;
- the redistribution policy of the running tasks under PE failure;
- the preemption policy of each task.

With reference to the first point, we assume that the PE's do not interact with each other, so that the processing capacity available in state \( i \) can be assumed proportional to the number of working PE's (i.e \( r_i = i \cdot r \), being \( r \) the capacity of a single PE). However, the non-interacting case can be considered as an upper bound since contention on shared resources or on communication channels reduces the achievable computation capacity in actual architectures. More realistic figures can be derived from detailed performance analysis of actual configurations, such as single bus [16] or multiple bus [17,18] architectures.

We concentrate on a particular case with \( n = 2 \) PE's and \( \nu = 2 \) tasks. The completion time problem is represented in Figure 4. Task 1 (2) is executed in state 2 at a rate \( r_{21} \) (\( r_{22} \)) and in state 1 at a rate \( r_{11} \) (\( r_{12} \)). When the system is in the failed state 0 no useful computation can be done, thus \( r_{01} = r_{02} = 0 \). \( X_1 (X_2) \) is the work requirement of task 1 (2). Absorption into the state labelled \( \text{COMPL1} \) means that Task 1 has completed its work requirement first, while absorption into the state labelled \( \text{COMPL2} \) means that Task 2 has completed first.

Furthermore, in order to compare different loading disciplines, 4 different cases are discussed:

Case 1: Both tasks are \( \text{prs} \); load sharing is uniform and load redistribution under failure is uniform. PE's are assigned uniformly to the running tasks.

Case 2: Task 1 is \( \text{prs} \) while Task 2 is \( \text{prd} \). Load sharing and load redistribution are uniform as in Case 1 above.
The uniform redistribution of the load, in cases 1 and 2, is translated into the MRMM by setting \( r_{21} = r_{22} = r \) and \( r_{11} = r_{12} = r/2 \). The effect of the system degradation is to slow down the execution of both tasks.

**Case 3:** Both tasks are **pr**. Load sharing is uniform in state 2, while load redistribution assigns higher priority to task 1: in state 1 only task 1 is carried on.

**Case 4:** Tasks 1 is **pr** while Task 2 is **prd**. Load sharing and load redistribution are defined as in Case 3 above: in state 1 only task 1 is carried on.

Cases 3 and 4 give rise to the following reward rates:

\[
\begin{align*}
r_{21} &= r_{22} = r \\
r_{11} &= r, \quad r_{12} = 0
\end{align*}
\]

The execution rate of Task 1 is insensitive to a single PE failure, while the same failure event stops the execution of Task 2.

The Laplace transform of the probability distribution function \( F(t) \) that Task 1 (Task 2) completes at time \( t \) before Task 2 (Task 1) can be computed from [1]. However, the knowledge of the multiple Laplace transform does not provide an easy way to get a numerical solution in the time domain [19]. We follow an alternative approach. If the work requirements \( X_1 \) and \( X_2 \) are \( PH \) random variables [10], the distribution of the completion time is also \( PH \) and, hence, can be computed as the absorption probability in a suitably generated Markov chain. The generation of the equivalent Markov chain is discussed in [11] for a single task and generalized in [1] for multiple tasks.

With reference to Figure 4, we have assumed the following set of numerical values:

\[
\begin{align*}
n &= 0.9 \\
\lambda &= 1 \cdot 10^{-3} h^{-1} \\
\mu &= 1 h^{-1}
\end{align*}
\]

Moreover, the initial probability vector \( P^0 \) is assumed to be equal to the steady-state probability vector of \( Z(t) \). Finally, the work requirements \( X_1 \) and \( X_2 \) are assumed to be Erlang distributed with order \( m \) (m stages) and expected values (in arbitrary work units) \( E[X_1] = 5 \) and \( E[X_2] = 10 \).

In order to investigate the influence of the variance of the work requirement distribution on the minimal completion time of the parallel tasks we have run the example of Figure 4 with two values of \( m \) (\( m = 2 \) and \( m = 10 \)). The distribution functions \( F(t) \) (absorption in \( COMPL_1 \)) and \( F(t) \) (absorption in \( COMPL_2 \)), are compared in Figure 5 with the reward rate matrix defined from Case 1. A decrease in the variance of the work requirement distribution enhances the completion probability of the shortest task. The asymptotic completion time probability of Task 1 \( \{ F(\infty) = \lim_{t \to \infty} F(t) \} \) is reported in Table I for the four previously described cases and for \( (m = 2, 10) \). Note that as \( t \to \infty \), \( F(t) + F(t) \to 1 \); completion of either tasks becomes the certain event.

![Figure 5 - Comparison of 1F(t) and 2F(t) in Case 1](image)

The probability of completing Task 1 before Task 2 increases passing from Case 1 to Case 2 and from Case 3 to Case 4, since the assumed reward model makes completion of Task 2 more difficult. Of course, the converse holds for Task 2. The largest asymptotic completion probability for Task 1 is obtained in Case 4, while the largest asymptotic completion probability for Task 2 is obtained in Case 1.

### Table I

<table>
<thead>
<tr>
<th>Case</th>
<th>Case 2</th>
<th>Case 3</th>
<th>Case 4</th>
</tr>
</thead>
<tbody>
<tr>
<td>( m = 2 )</td>
<td>0.7407</td>
<td>0.7488</td>
<td>0.7459</td>
</tr>
<tr>
<td>( m = 10 )</td>
<td>0.9352</td>
<td>0.9415</td>
<td>0.9385</td>
</tr>
</tbody>
</table>
5 Conclusion

A Multi-Reward Stochastic Model is defined with the aim of representing the parallel execution of multiple tasks in a randomly varying system. The execution (reward) rates, to be incorporated into the model, can be borrowed from more detailed performance studies.

The distribution of the minimal completion time can be computed in the Laplace transform domain or as the absorption probability in a properly generated Markov chain when the work requirements are PH random variables.

Various physical interpretations of the MRMM are discussed, and a numerical example is provided.

References


